Administration Exercises

Bruce Momjian

February 2004

Installation

1. Run configure
2. Compile
3. Install
4. Run initdb
5. *Set the super-user password during initdb
6. *Set a server encoding during initdb
7. Start postmaster
8. Use pg_controldata to see current server settings

---

9. Create several databases
10. *Set up the postmaster to automatically start on reboot, and test
11. *Configure the postmaster to automatically stop on system shutdown, test
12. *Run multiple postmasters

Access Control

1. Create several database users
2. Connect to a database as several users
3. Restrict connections to a single user
4. Restrict connections to a single database
5. Use socket permissions to restrict local access
6. Create a password for a user
7. Restrict access to a specific group
8. Enable and test MD5 encryption
9. Enable internet connections
10. *Enable plain password and snoop on-wire password
11. Enable access to 172.20.30.14

12. Enable access to 172.20.31/24
13. Enable access to 172.20.63.80/30
14. *Create an SSL connection and force SSL connections
15. *Enable local ident or host ident connections
16. *Create a mapping from Unix names to database names
17. *Enable per-db users
18. *Add an object to template1 and verify it appears in a new database

Server Configuration

1. Increase the maximum number of connections
2. Increase the number of shared buffers
3. *Increase the kernel shared memory limit to support more shared buffers
4. *Start another postmaster on a different port and connect to it

---

5. Enable full logging output in the server logs
6. Enable full logging output to the client
7. Print only statements that generate errors

---

8. Print all queries
9. Print queries that take over several seconds
10. Print timestamps on every log line
11. Start two sessions and grep specific sessions from the logs
12. Find the username of a that session
13. *Enable syslog server reporting
14. *Rotate the server log file using rotatelogs
15. Cancel queries taking more than a few seconds

---

16. Create a database, create some tables, dump the database, drop the database, recreate the database, reload the database
17. Same as above, but reload into a different database

---

18. Install an interface from http://gborg.postgresql.org
19. Install a /contrib module

---

20. Create several databases, dump them all into a single file, initdb and reload
21. *Install PostgreSQL-enabled PHP, if not already present, test

Resource Management

1. Determine the write-ahead log file disk usage
2. Generate checkpoint-too-frequent log message
3. *Increase WAL file disk usage
4. Move WAL to another disk (or file system if only one disk)
5. *Prove that WAL will preserve data after an OS crash
6. Schedule automatic backups
7. Schedule VACUUM, VACUUM FULL, and ANALYZE

Monitoring

1. Enable statistics collector
2. Print active queries from pg_stat_activity
3. Run a long query and find it using ps, analyze cpu and I/O used
4. Monitor database activity and disk space during the regression test
5. Display all currently running queries
6. Display cpu and memory usage of a specific database session
7. Find the actual file used by a table
8. Verify that VACUUM doesn't shrink the table size, but VACUUM FULL does
9. Find the file name of a TOAST table
10. Use oid2name to verify the file name
11. Find the table taking the most disk space
12. Find the largest table but include TOAST and indexes
13. Estimate disk space for a table and test your estimate
14. Find the total disk space for a each database
15. *Move a database to a different drive
16. *Move a table or index to another drive